```
%{
#include <stdio.h>
extern FILE *yyin;
extern char *yytext;
extern int yylineno;
%}

%token START FINISH DEF NUMBER STRING CHAR ARRAY OF UNDEFINED READ IF STARTIF
FINISHIF ASSIGN
%token WHILE STARTWHILE FINISHWHILE PROC STARTPROC FINISHPROC CALL RETURN
PRINT
%token id constant

%token NEGPOSDIGIT
%token ERRORNUMCONST
%token OP_PLUS OP_MINUS OP_MUL OP_DIV OP_LT OP_LTE OP_EQ OP_NEQ OP_RT OP_RTE
OP_OR OP_AND
%token SEP_SEMICOL SEP_COM SEP_COL SEP_SQBR SEP_SQBREND SEP_RBR SEP_RBREND

%%
program: START cmds FINISH
        ;
cmds: cmd cmdsconf
        ;
cmdsconf: /*epsilon*/
        | cmds
        ;
cmd: simplecmd
        | structcmd
        ;
simplecmd: defcmd
        | assigncmd
        | readcmd
        | printcmd
        | returncmd
        ;
defcmd: DEF declist
        ;
declist: declaration declistconf
        ;
declistconf: /*epsilon*/
        | SEP_SEMICOL declist
        ;
declaration: id SEP_COL dtype
        ;
dtype: primitive
        | arraydecl
        ;
```

```
primitive: NUMBER
        | STRING
        | CHAR
        ;
arraydecl: ARRAY SEP_SQBR arraydeclconf
        ;
arraydeclconf: constant SEP_SQBREND OF primitive
        | id SEP_SQBREND OF primitive
        ;
assigncmd: ASSIGN id SEP_COL assigncmdconf
        ;
assigncmdconf: symbolvalue
        | SEP_RBR expressionstart SEP_RBREND
        | UNDEFINED
        ;
symbolvalue: id symbolvalueid
        | constant
        | SEP_SQBR symbolvalueconf
        ;
symbolvalueid: /*epsilon*/
        | SEP_SQBR symbolvalueconf
        ;
symbolvalueconf: id SEP_SQBREND
        | constant SEP_SQBREND
        ;
expressionstart: term expression
        ;
expression: OP_PLUS term expression
        | OP_MINUS term expression
        | /*epsilon*/
        ;
term: factor muldiv
        ;
muldiv: OP_MUL factor muldiv
        | OP_DIV factor muldiv
        | /*epsilon*/
        ;
factor: SEP_RBR expressionstart SEP_RBREND
        | symbolvalue
        ;
readcmd: READ id readcmdconf
        ;
readcmdconf: /*epsilon*/
        | SEP_SQBR symbolvalueconf
        ;
printcmd: PRINT SEP_RBR expressionprint SEP_RBREND
        ;
expressionprint: factorprint expressionprintconf
```

```
        ;
expressionprintconf: /*epsilon*/
        | OP_PLUS expressionprint
        ;
factorprint: id
        | constant
        | callstmt
        ;
returncmd: RETURN returncmdconf
        ;
returncmdconf: expressionstart
        | callstmt
        ;
structcmd: ifstmt
        | whilestmt
        | procstmt
        | callstmt
        ;
ifstmt: IF condition STARTIF cmds FINISHIF
        ;
condition: basiccondition conditionconf
        ;
conditionconf: /*epsilon*/
        | logicaloperator condition
        ;
basiccondition: symbolvalue comparisonoperator basicconditionconf
        ;
basicconditionconf: symbolvalue
        | UNDEFINED
        ;
comparisonoperator: OP_LT
        | OP_RT
        | OP_LTE
        | OP_RTE
        | OP_EQ
        | OP_NEQ
        ;
logicaloperator: OP_AND
        | OP_OR
        ;
whilestmt: WHILE condition STARTWHILE cmds FINISHWHILE
        ;
procstmt: PROC id SEP_RBR procstmtconf
        ;
procstmtconf: SEP_RBREND STARTPROC cmds FINISHPROC
        | declist SEP_RBREND STARTPROC cmds FINISHPROC
        ;
callstmt: CALL id SEP_RBR paramslist SEP_RBREND
```

```
        ;
paramslist: expressionstart paramslistconf
        | /*epsilon*/
        ;
paramslistconf: /*epsilon*/
        | SEP_COM paramslist
        ;
%%
int main(int argc, char **argv)
{
    if (argc == 2) {
        yyin = fopen(argv[1], "r");
        yyparse();
    }
    else{
        printf("No input file given!\n");
    }
    if(0==yyparse()) printf("Result yyparse OK");
}

int yyerror(char *s)
{
    printf("Error on line #%d\n", yylineno);
    printf("Unexpected token: '%s'\n", yytext);
    return 0;
}
```